Aplikasi Pencarian Pekerja Jasa Rumah Tangga di Sekitar dengan Metode Radius dan Rating Berbasis Android

Jerry Erlangga^{#1}, Eliyani^{#2}

[#]Program Studi Informatika Fakultas Ilmu Komputer, Universitas Mercu Buana
Jl. Meruya Selatan No.1 Kembangan, Jakarta 11650

¹jerryerlangga82@gmail.com

²eliyani@mercubuana.ac.id

Abstrak—Berkurangnya lapangan pekerjaan akibat penurunan beberapa sektor usaha dikarenakan masa krisis akibat pandemi, membuat banyak pekerja memilih beralih menjadi pekerja rumah tangga untuk memperoleh penghasilan. Penelitian ini bertujuan untuk membuat aplikasi yang dapat mempertemukan pencari kerja dengan pemberi kerja. Digunakan Metode Radius yang memanfaatkan *Google Maps API* dalam mengakses lokasi pemberi kerja dan pencari kerja sehingga pencari kerja dapat dengan mudah mencari pekerjaan yang berada disekitarnya. Pekerjaan yang telah diselesaikan akan diberikan *rating* oleh pemberi kerja sesuai tingkat kepuasan terhadap hasil pekerjaan pencari kerja. Aplikasi yang dibangun sudah memiliki peningkatan keamanan seperti enkripsi koneksi *web service* ke server *Backend*, verifikasi OTP nomor telepon, dan verifikasi data pengguna dengan foto kartu identitas. Menggunakan metode *black box testing*, semua fitur dapat berjalan secara optimal. Pengujian dilakukan terhadap 20 pengguna dengan menggunakan mobile phone berbasis android dengan beragam versi. Hasil ujicoba menunjukkan bahwa semua fitur dapat berjalan secara optimal pada Android versi 7 hingga 11 dengan akurasi 95% namun 1 dari 20 pengguna tersebut tidak dapat menjalankan aplikasi karena android yang digunakan berada di bawah versi 7.

Kata kunci— Pekerjaan Rumah Tangga, Radius, Google Maps API, Rating, Android

I. PENDAHULUAN

Krisis ekonomi yang terjadi di berbagai negara saat ini menyebabkan banyak perusahaan mengalami ketidakstabilan keuangan, hal ini menimbulkan terjadinya pemutusan hak kerja besar-besaran yang dilakukan oleh beberapa perusahaan. Dari hal tersebut berimbas kepada banyaknya tenaga kerja yang kehilangan pekerjaan dan menyulitkan mereka untuk dapat mencari pekerjaan kembali. Sebagian dari mereka ada yang beralih ke bisnis usaha mikro kecil dan menengah ataupun bekerja menawarkan jasa mereka guna mencukupi kebutuhan sehari - hari.

Menurut *International Labour Organization (ILO)*, pada tahap awal diperkirakan bahwa sekitar 5,3 juta sampai 24,7 juta pekerjaan di dunia dapat hilang disebabkan oleh pandemi COVID-19[1]. Selanjutnya ILO memprediksi jam kerja seluruh pekerja akan terjadi penurunan pada kuartal kedua tahun 2020 yakni sebesar 10,5 persen atau setara dengan 305 juta penuh waktu dengan asumsi jam kerja penuh waktu dalam seminggu adalah 48 jam [2]. Dikutip dari situs kementerian ketenagakerjaan, berdasarkan dari hasil survei angkatan kerja nasional periode Agustus 2020 yang dilakukan oleh BPS, diketahui bahwa sekitar 14,28 persen dari total penduduk usia kerja di Indonesia terdampak akibat pandemi Covid-19, di antaranya sekitar 2.562.530 orang atau 1,26 persen yang menjadi pengangguran, kemudian sekitar 757.574 orang atau 0,37 persen yang keluar dari pasar kerja, dan sebanyak 1.772.760 orang atau 0,87 persen yang sementara tidak bekerja, serta 24.027.576 orang atau 11,78 persen yang mengalami pengurangan jam kerja [3].

Berdasarkan keterangan dari Aboy selaku pemilik yayasan penyalur asisten rumah tangga yang bernama Purwa Sentosa Agency di Tangerang, beliau membenarkan bahwa banyak permintaan bekerja sebagai Asisten Rumah Tangga (ART) datang dari para korban Pemutusan Hak Kerja (PHK). Tingginya permintaan tersebut tidak sebanding dengan jumlah permintaan jasa rumah tangga yang beliau dapat, sehingga tidak semua korban PHK tersebut disalurkan menjadi ART [4].

Permintaan akan pekerjaan berupa jasa rumah tangga dapat ditingkatkan dengan disediakannya suatu aplikasi yang dapat mengakomodir untuk mempertemukan calon pemberi kerja dengan pencari kerja. Aplikasi ini dijalankan pada *smartphone* karena masyarakat dapat lebih mudah dan fleksibel dalam mengakses berbagai aplikasi maupun mencari informasi melalui *smartphone* terutama yang berbasis android. Penunjang *smartphone* pun kini semakin berkembang cepat dengan bermacam-macam fitur dan juga fungsinya [5]. Melalui *smartphone* dapat diakses layanan

lokasi dan bertukar informasi melalui internet sehingga mempermudah dalam perancangan aplikasi. Menurut kabar dari Buletin APJII Edisi 74 November 2020, *smartphone* merupakan perangkat favorit pengguna internet di Indonesia dengan jumlah 95,4% kemudian disusul oleh pengguna perangkat laptop atau tablet sebanyak 19,7%, dilanjutkan dengan pengguna komputer PC sebanyak 9,5% [6].

Penelitian sebelumnya telah menghasilkan aplikasi pencarian pekerjaan paruh waktu yang hanya menampilkan jarak dari lokasi pekerjaan tanpa menggunakan pembacaan radius yang terdekat dari lokasi pengguna [7]. Pada penelitian lainnya yang menggunakan metode radius dan *rating* diterapkan pada lokasi wisata di mana pada aplikasi tersebut hanya menyajikan lokasi wisata berdasarkan metode radius dan rating [8]. Pada jurnal ini aplikasi yang dibangun menerapkan metode radius dan rating untuk keperluan pencarian pekerjaan rumah tangga di mana dari proses pembuatan pekerjaan sampai pekerjaan tersebut diselesaikan oleh pencari kerja dapat dikelola dalam aplikasi.

Perancangan aplikasi ini memanfaatkan *Google Maps API* dalam mengambil lokasi dan menampilkan peta. Lokasi yang didapat disimpan ke dalam database dalam bentuk koordinat *longitude* dan *latitude* yang digunakan sebagai acuan dalam mendeteksi pekerjaan yang berada di sekitar pekerja. Ketika pekerjaan yang telah dibuat oleh pemberi kerja telah diambil oleh pencari kerja, maka pemberi kerja mendapatkan informasi berupa notifikasi bahwa pekerjaan tersebut sudah mendapatkan pekerja.

II. METODE PENELITIAN

Tahapan metode penelitian yang dilakukan untuk merancang aplikasi pencarian pekerja jasa rumah tangga di sekitar dengan metode radius dan rating berbasis Android menggunakan metode *SDLC (Software Development Life Cycle)* dengan model *Waterfall*. Metode *SDLC Waterfall* (lihat Gambar 1) adalah metode pengembangan perangkat lunak dengan cara berurutan, yakni seluruh proses pengerjaan tahapan yang ada dikerjakan secara berurutan yang diilustrasikan terus mengalir ke bawah seperti air terjun [9].

Gambar. 1 Metode SDLC : *Waterfall Model*

A. Analisis Kebutuhan

Dalam membantu perancangan aplikasi ini, dilakukan pengambilan data kepada 51 responden secara acak melalui

kuesioner yang terdiri dari mahasiswa, karyawan swasta, ibu rumah tangga dan masyarakat yang belum memiliki pekerjaan. Pada Gambar 2, hasil kuesioner menunjukkan bahwa sebanyak 72,5% responden berpendapat bahwa keterbatasan jumlah permintaan pekerjaan dibandingkan dengan pelamar merupakan hal yang mempersulit mereka dalam mencari kerja. Jenis pekerjaan yang paling diminati berdasarkan hasil kuisisioner merupakan membersihkan rumah dengan nilai 60% diikuti oleh mencuci atau menyetrrika pakaian dengan nilai 58% yang dapat dilihat pada Gambar 3. Aplikasi pencari pekerja jasa rumah tangga ini lebih diminati oleh responden jika dapat dijalankan pada aplikasi berbasis Android dengan nilai 80,4% dari pada berjalan pada *website* dengan nilai 19,6% yang dapat dilihat pada Gambar 4.

Gambar. 2 Hal yang mempersulit dalam mencari pekerjaan

Gambar. 3 Jasa yang paling diminati

Gambar. 4 Platform yang paling diminati

B. Desain Sistem

Setelah fitur-fitur aplikasi berhasil didapatkan pada proses analisa, tahapan selanjutnya adalah *Design Sistem*. Pada tahapan ini, aplikasi dirancang berjalan pada perangkat *smartphone* berbasis Android. Android merupakan sistem operasi berbasis Linux buatan Google yang dirancang untuk keperluan perangkat bergerak yang menggunakan layar sentuh seperti *smartphone* dan

komputer tablet [10]. Fitur yang ada pada aplikasi ini digambarkan dalam *Unified Modeling Language (UML)*. *UML* merupakan salah satu tools yang biasa dipergunakan dalam merancang pengembangan software yang berbasis *object-oriented*. *UML* memiliki standar penulisan sebuah sistem *blueprint* yang di antaranya meliputi konsep proses bisnis, skema *database*, penulisan kelas-kelas dalam bahasa program yang spesifik, dan komponen yang dibutuhkan dalam membangun sistem perangkat lunak [11]. Bentuk *UML* yang akan dibuat pada tahapan ini antara lain *Use Case Diagram*, *Class Diagram* dan *Activity Diagram*.

C. Implementasi

Metode radius digunakan dalam mencari pekerjaan yang memiliki rentang jarak maksimal yang telah di atur didalam aplikasi seperti disajikan pada Gambar 5.

Gambar. 5 Penerapan metode radius

Perhitungan jarak dari lokasi pekerjaan ke lokasi pencari kerja berdasarkan koordinat *latitude* dan *longitude* dengan menggunakan rumus *Haversine* yang pada aplikasi ini dibantu dengan library dari *Google Maps SphericalUtil* untuk membantu dalam menghitung jaraknya sehingga tidak dilakukan perhitungan secara manual. Perangkat Android mengambil koordinat dari perangkat pencari kerja kemudian melakukan perhitungan jarak ke lokasi pekerjaan. Pengambilan koordinat pada aplikasi menggunakan *Global Positioning System (GPS)* dari *Google Maps*. Layanan *Google Maps* menyediakan *API (Application Programming Interface)* yang dapat digunakan oleh developer lain dalam menerapkan peta di aplikasi buatannya [12]. Koordinat pekerjaan yang masuk ke dalam radius pencari kerja ditampilkan dalam halaman maps dalam bentuk *marker*. Penggunaan *GPS* memudahkan pengembang memanfaatkan nilai-nilai geografis yang berupa koordinat *latitude* dan *longitude* untuk mengembangkan aplikasi-aplikasi yang berbasis *LBS (Location Based Service)* sehingga dapat memberikan informasi tentang posisi pengguna, hal terdekat yang berada di sekitar pengguna, serta rute menuju suatu letak atau lokasi [13]. *LBS* memiliki fungsi untuk mengidentifikasi titik lokasi dari seseorang atau objek tertentu, seperti menemukan lokasi tempat ibadah terdekat atau lokasi lainnya [14]. Pada aplikasi ini layanan *LBS* digunakan untuk membaca titik lokasi pencari kerja dan lokasi pekerjaan yang dibuat oleh pemberi kerja.

Metode *rating* digunakan sebagai pengukur kinerja dari pencari kerja. Nilai *rating* diberikan oleh pemberi kerja ketika pekerjaan telah diselesaikan oleh pencari kerja. *Rating* pencari kerja didapat berdasarkan nilai rata-rata dari total *rating* yang didapat dari pekerjaan yang telah diselesaikannya. Nilai *rating* yang didapat disimpan oleh

Backend. Penerapan metode *rating* dalam aplikasi ini disajikan pada Gambar 6.

Gambar. 6 Penerapan metode rating

Backend pada aplikasi ini merupakan sebuah server berbasis *cloud* yang berguna sebagai jembatan antara antarmuka aplikasi Android ke database. *Backend* dibuat dengan bahasa pemrograman Java serta menggunakan framework *Spring Boot* yang dalam komunikasinya saling bertukar data dengan aplikasi Android menggunakan *web service* berformat *JavaScript Object Notation (JSON)*. *Web service* adalah suatu kumpulan layanan berbasis web yang menggunakan jaringan protokol HTTP. *Web service* dapat diakses dan dimanfaatkan oleh pengguna dengan bahasa pemrograman, arsitektur dan sistem operasi yang berbeda (*interoperability*) [15].

D. Pengujian Program

Setelah aplikasi selesai dibuat, tahap selanjutnya adalah pengujian program. Pada tahap ini dilakukan pemeriksaan terhadap semua fitur dan fungsi yang ada dalam aplikasi sesuai dengan rancangan yang telah ditentukan. Tahap pengujian aplikasi menggunakan metode *Black Box*. *Black Box Testing* merupakan suatu metode pengujian aplikasi yang membahas melalui sisi luar suatu aplikasi perangkat lunak, yakni dari mulai tampilan hingga aksi input data [16]. Pengujian pada aplikasi ini hanya difokuskan terhadap aksi dan fungsi dari aplikasi sesuai dengan rancangan tanpa memperdulikan bagaimana cara aplikasi melakukannya. Dengan itu maka penguji perlu mengetahui apa yang akan diujinya pada aplikasi tersebut.

III. HASIL DAN PEMBAHASAN

Aplikasi Pencarian Pekerja Jasa Rumah Tangga di Sekitar dengan Metode Radius dan Rating Berbasis Android memiliki koneksi dengan *Backend* dan *Firebase*. *Backend* merupakan suatu program yang berjalan di latar belakang yang mendukung dalam pemrosesan suatu aplikasi. *Backend* pada rancangan ini menggunakan framework *Spring Boot* dengan bahasa pemrograman Java. *Backend* digunakan dalam berkomunikasi dengan aplikasi Android melalui *web service*, di mana setiap perintah yang diterima dari aplikasi akan diproses di *backend* kemudian disimpan ke database. Dalam dunia teknologi, penggunaan *web service* dengan metode *Representational state transfer (RESTfull)* merupakan standar saat ini pada urusan penyediaan data dari *Backend* ke aplikasi [17]. Selain berkomunikasi dengan *backend*, aplikasi ini juga berkomunikasi dengan beberapa produk yang terdapat dalam *Google Firebase*. *Firebase* adalah suatu penyedia layanan *cloud* dari Google yang didirikan oleh Andrew Lee dan James Tamplin pada tahun 2011 [18]. *Firebase* memiliki beberapa produk yang mampu mempermudah *developer* dalam mengembangkan aplikasi.

Aplikasi ini menggunakan *Firebase Authentication* sebagai validasi pengguna ketika hendak masuk ke dalam aplikasi. Pengguna akan diminta memasukkan nomor telepon yang nantinya *Firebase Authentication* akan mengirim *Short Message Service (SMS)* berisi *One-Time Password (OTP)* ke nomor tersebut lalu pengguna memasukkan *OTP* yang diterima agar dapat masuk ke dalam aplikasi. Selain itu, pada aplikasi ini juga memanfaatkan *Firebase Realtime Database* dan *Cloud Messaging* untuk memberikan notifikasi ke perangkat pengguna pada saat transaksi hingga pemberian ulasan. *Firebase Realtime Database* adalah suatu basis data *online* yang digunakan untuk menyimpan data dari aplikasi dalam bentuk format *JSON* dan dapat disinkronkan secara *realtime* ke setiap *client* yang terhubung ke *Firebase* [19]. *Firebase Cloud Messaging (FCM)* merupakan layanan yang dapat diakses pada semua jenis perangkat (*cross-platform*) yang digunakan dalam berkirim pesan secara gratis yang disediakan oleh Google. *FCM* memiliki fungsi untuk melakukan *push notification*, yakni notifikasi yang muncul di bagian atas layar perangkat *smartphone* [20]. Perancangan Arsitektur Sistem untuk aplikasi ini dapat dilihat pada Gambar 7.

Gambar. 7 Perancangan arsitektur sistem

Salt = z0Dn0YvaffveRMVYVEEMO9DtohVlyZIWN5xzi
 Securitycode = Sha-256 (Salt + sessionID + timestamp)

Contoh Request :

sessionID	29
timestamp	12345678
securitycode	ae2d72a61aedd4f4f48d63cae6afdd70402d73c4074c4f5d5c44281432705e9

Gambar. 8 Enkripsi Web Service

Aplikasi ini diakses oleh tiga pengguna dengan hak akses yang berbeda yakni admin yang melakukan verifikasi kartu identitas pengguna, pemberi kerja yang membuat permintaan pekerjaan ke dalam aplikasi serta pencari kerja yang bertindak sebagai pekerja yang mengambil pekerjaan. Pengolahan data dari pengguna disimpan ke *database* melalui *Backend*. *Database* yang digunakan dalam perancangan aplikasi ini menggunakan *PostgreSQL* yakni suatu sistem penyimpanan basis data *open source* yang sangat baik dalam menangani data dalam jumlah besar dibanding dengan *Mysql* [21]. Setiap *web service* yang terjadi antara aplikasi dan *Backend* dilindungi dengan mencocokkan kode keamanan (*security code*) berdasarkan

hasil enkripsi dari aplikasi dan *Backend*, jika kode keamanan tidak cocok maka proses transaksi akan ditolak. Bentuk enkripsi yang digunakan pada aplikasi ini dapat dilihat pada Gambar 8.

Salt merupakan suatu teks berupa karakter acak yang digunakan untuk menyulitkan peretasan keamanan. Biasanya *salt* digunakan pada metode enkripsi *hash* sebagai tambahan input. Dengan penambahan *salt*, penyerangan keamanan akan lebih sulit karena harus mengetahui nilai *salt* dan menambahkannya ke setiap enkripsi. Selain itu, karena pada penerapan *Salt* pada fungsi *hash* membuat karakter sandi memiliki panjang yang sama, *Salt* dikatakan mampu menjaga karakter pada kata sandi [22]. *Salt* memiliki nilai yang sama serta disimpan pada aplikasi dan *Backend*. Sedangkan nilai *timestamp* didapatkan dari waktu transaksi *web service* terjadi. Nilai *sessionID*, *timestamp* dan *securitycode* akan selalu ada pada setiap transaksi yang berlangsung antara aplikasi dan *Backend* untuk memastikan data yang dikirimkan adalah valid dan mengurangi resiko dari peretasan. *Backend* terhubung dengan *database* sebagai media penyimpanan data. *Database* yang digunakan pada perancangan aplikasi ini menggunakan *PostgreSQL*.

Perancangan *database* pada aplikasi ini ditunjukkan pada Gambar 9. Tabel *m_user* merupakan tabel yang berfungsi untuk menyimpan data pengguna yang mendaftar ke dalam aplikasi, di mana tabel ini memiliki relasi ke tabel *m_user_session*. Tabel *m_user_session* berfungsi dalam menyimpan *session* dari *user* ketika berada dalam aplikasi.

Gambar. 9 Class diagram

Tujuan penggunaan *session* adalah agar tidak terjadi *multi login* yakni keadaan di mana satu akun pengguna digunakan pada beberapa perangkat berbeda secara bersamaan. Pada tabel *m_order* berfungsi menyimpan transaksi dan data dari pekerjaan yang telah dibuat oleh pemberi kerja kedalam aplikasi. Tabel *m_order* memiliki pemicu (*trigger*) ke *m_order_history* di mana setiap kali ada perubahan data yang terjadi pada tabel *m_order* maka data tersebut akan tersimpan ke dalam tabel *m_order_history*. Tujuan penggunaan tabel *history* ini adalah sebagai audit agar admin aplikasi dapat mengetahui rekam jejak perubahan transaksi ketika mengalami suatu masalah.

Pengguna dalam aplikasi ini memiliki 3 hak akses yang berbeda. Hak akses pada aplikasi ini dibagi menjadi 3 yakni sebagai Pencari Kerja, Pemberi Kerja atau Admin seperti ditunjukkan pada Gambar 10.

Gambar. 10 Use Case diagram

Pemberi kerja merupakan pengguna yang memiliki akses untuk membuat permintaan pekerjaan ke dalam aplikasi, sedangkan pencari kerja adalah pengguna yang memiliki akses dalam mencari dan mengambil pekerjaan yang sudah dibuat oleh pemberi kerja. Pekerjaan diseleksi berdasarkan radius jarak yang diatur oleh pencari kerja ketika melakukan pencarian pekerjaan. Pekerjaan yang jaraknya berdekatan dengan titik koordinat dari pencari kerja, akan terlihat pada aplikasi dengan tanda berupa *marker* berwarna merah pada tampilan peta. Admin berperan dalam melakukan verifikasi data pencari kerja dan pemberi kerja yang mendaftar kedalam aplikasi.

Data yang diverifikasi berupa data kartu identitas seperti Kartu Tanda Penduduk (KTP) serta foto pengguna dengan posisi memegang kartu identitas di tangannya. Hal ini diperlukan untuk mengurangi adanya tindak penyalahgunaan pada aplikasi. Setelah data berhasil di verifikasi dan dinyatakan valid maka user dan worker dapat melakukan aktifitas di dalam aplikasi. Aktifitas pencari kerja dan pemberi kerja dalam penggunaan aplikasi dapat dilihat pada Gambar 11.

Gambar. 11 Activity Diagram Login

Pengguna aplikasi mulanya melakukan pendaftaran dengan mengisi data nama lengkap, nomor telepon dan password serta memilih akses akan mendaftar sebagai pemberi kerja atau pencari kerja. Jika pengguna memilih mendaftar sebagai pemberi kerja, maka pengguna tersebut hanya dapat memasukkan pekerjaan ke dalam aplikasi, dan jika mendaftar sebagai pencari kerja maka pengguna tersebut hanya dapat mengambil pekerjaan yang tersedia pada aplikasi. *User Interface (UI)* dari halaman masuk dan daftar dapat dilihat pada Gambar 12.

Gambar. 12 UI masuk dan daftar

Ketika pengguna hendak masuk ke dalam aplikasi, pengguna akan diminta memasukkan kode berupa *OTP* yang dikirimkan ke nomor telepon dimana hal ini sebagai

verifikasi bahwa nomor telepon yang diinput adalah benar milik pengguna. Pada menu ini terdapat tombol Kirim Ulang Kode OTP yang berguna dalam mengirimkan kembali kode OTP apabila SMS tersebut belum diterima pada kotak pesan di nomor telepon pengguna. Setiap pengguna yang menekan tombol verifikasi maka dianggap menyetujui peraturan dari penggunaan aplikasi yang dapat kita sematkan berupa halaman persetujuan pengguna ketika menekan kata Syarat dan Ketentuan yang berada diatas tombol verifikasi. UI dari verifikasi OTP dapat dilihat pada Gambar 13. Jika verifikasi berhasil maka pengguna akan diarahkan ke halaman dashboard aplikasi.

Gambar. 13 UI Verifikasi OTP

Tahapan selanjutnya setelah berhasil masuk ke dalam aplikasi, pengguna aplikasi diminta untuk melakukan aktivasi akun mereka. Tahapan aktivasi akun dapat dilihat pada Gambar 14. Pencari kerja dan pemberi kerja yang telah mendaftar ke dalam aplikasi diwajibkan untuk melakukan verifikasi data dengan mengunggah foto kartu identitas berupa KTP serta foto diri dengan posisi memegang kartu identitas tersebut. Pemberian verifikasi seperti ini memang tidak menjamin akan tidak adanya tindak penyalahgunaan di dalam aplikasi tetapi setidaknya dapat mengurangi adanya tindak penggunaan tersebut. UI verifikasi kartu identitas dapat dilihat pada Gambar 15. Pada saat pertama kali pengguna ataupun mitra mendaftar ke aplikasi, maka status mereka adalah “Belum diverifikasi” sehingga mereka tidak dapat mengakses untuk membuat pekerjaan ataupun mengambil pekerjaan. Setelah mereka mengunggah kartu identitas, status mereka akan berubah menjadi “Dalam Peninjauan”. Data yang diunggah pengguna akan disimpan ke *Firebase Storage*, kemudian admin aplikasi akan melakukan verifikasi data identitas pengguna. Admin akan mencari pengguna dengan status “Dalam peninjauan” lalu melihat gambar yang diunggah tersebut. Setelah data dinyatakan *valid* maka admin akan mengubah status pengguna menjadi “Terverifikasi”. Jika

data yang diunggah tidak sesuai maka admin akan mengubah statusnya menjadi “Ditolak”, yang nantinya pengguna atau mitra diwajibkan mengunggah foto identitas ulang hingga admin melakukan peninjauan data tersebut seperti tahap sebelumnya. Halaman verifikasi pengguna yang dilihat oleh admin ditunjukkan pada Gambar 16.

Gambar. 14 Activity diagram aktivasi akun

Gambar. 15 UI Unggah kartu identitas

Pengguna yang sudah terverifikasi dapat membuat permintaan pekerjaan di aplikasi. Tahapan pembuatan pekerjaan dapat terlihat pada Gambar 17. Dalam membuat pekerjaan, pemberi kerja akan diminta memilih titik koordinat lokasi kerja kemudian mengisi data pelengkap

sesuai yang diminta pada aplikasi. Halaman pengisian data pekerjaan yang akan diunggah dapat terlihat pada Gambar 18. Data yang diminta di antaranya meliputi kategori pekerjaan, nama pekerjaan, anggaran, durasi pekerjaan beserta alamat pekerjaan dengan titik koordinat.

Veemo Administrator			Akun Saya >
U	User 7	fM6fU8cEX2RE3XuCI3GJVMzW5I2	User ID 15
	Terverifikasi		
U	User 8	5MladihQCVzLszQaIdjci1Aedd2	User ID 14
	Terverifikasi		
W	Worker 4	efh7ElajEecFsDGUsVOTzQnUYw22	User ID 18
	Terverifikasi		
U	User 6	HsELdEilMRZ65RKvX0ckDILGE442	User ID 16
	Terverifikasi		
W	Worker 1	holzBsOg7jPnmHndU3qqN3SJdy2	User ID 20
	Terverifikasi		
W	Worker 3	IHQuwCm6FydSNZm6R2SezIgl14g1	User ID 19
	Terverifikasi		

Gambar. 16 Verifikasi data pengguna

Gambar. 17 Activity Diagram unggah pekerjaan

Kategori pekerjaan terdiri dari jenis pekerjaan yang akan dilakukan. Pada kolom nama pekerjaan dapat ditulis mengenai pekerjaan lebih terperinci, kolom upah merupakan informasi bahwa pengguna akan membayar pekerja sebesar nilai upah terhadap pekerjaan tersebut. Pada kolom pengisian alamat pekerjaan diisi secara otomatis dari data *Google Maps* berdasarkan titik koordinat yang dipilih sebelumnya.

The screenshot shows a mobile application interface for posting a job. It includes a map view on the left and a form on the right. The form fields are: 'Pilih Kategori Pekerjaan' (Laundry), 'Nama Pekerjaan' (Cuci Baju), 'Upah' (0), 'Durasi Pekerjaan' (Start Date: 18/06/2021, End Date: 18/06/2021), and 'Lokasi Pekerjaan' (Jl. Bumi Raya No.15, RT.7/RW.4, Duren Sawit, Kec. Duren Sawit, Kota Jakarta Timur, Daerah Khusus Ibukota Jakarta 13440, Indonesia). There is a 'Detail lokasi' section with a checkmark for 'Alamat Sudah Benar' and a 'LANJUTKAN' button. At the bottom right is an 'UNGGAH PEKERJAAN' button.

Gambar. 18 Unggah pekerjaan

Pekerjaan yang telah dibuat oleh pemberi kerja akan disimpan ke *database* sehingga pekerjaan tersebut dapat terlihat pada halaman utama serta dapat ditemukan oleh pencari kerja yang berada di sekitar lokasi dari titik kerja yang telah ditentukan. Pekerjaan yang dibuat pada aplikasi ini memiliki beberapa tahapan status. Pada saat pekerjaan dibuat statusnya adalah *Open* yang berarti dapat diambil oleh pencari kerja di sekitar. Pemberi kerja dapat melihat nama pekerja yang mengambil pekerjaan tersebut pada halaman detail pekerjaan. Selama pekerjaan tersebut belum diambil oleh pencari kerja, maka pekerjaan tersebut dapat dibatalkan oleh pemberi kerja. Pekerjaan yang dibatalkan akan berubah status menjadi *Cancel* dan tidak akan dapat ditemukan oleh pencari kerja.

Gambar. 19 Activity Diagram pencarian pekerjaan

Pada tahap selanjutnya merupakan aktivitas dari sisi pencari kerja dalam mencari pekerjaan yang berada di sekitarnya. Alur pencarian pekerjaan dapat dilihat pada Gambar 19. Pencarian pekerjaan pada aplikasi ini akan menghitung jarak dari titik koordinat pencari kerja dengan titik koordinat dari lokasi pekerjaan yang dibuat pemberi kerja kemudian jarak tersebut dihitung apakah masih termasuk kedalam jarak radius pencarian pekerjaan atau tidak.

Radius pada *Google Maps* adalah suatu cakupan wilayah dari suatu titik koordinat ke jarak yang telah ditentukan. Pada aplikasi pencarian ini radius dipergunakan untuk mendeteksi permintaan pekerjaan yang tersedia di sekitar lokasi koordinat pekerja sehingga hanya pekerjaan yang masuk ke radius terdekat yang akan muncul pada peta. Jarak jangkauan radius dari aplikasi ini dapat diatur oleh pengguna saat hendak melakukan pencarian pekerjaan di sekitarnya. Perhitungan jarak yang dilakukan pada aplikasi ini memanfaatkan *library* dengan potongan program sebagai berikut:

```
SphericalUtil.computeDistanceBetween(new
LatLng (x1,y1), new LatLng(x2,y2))
```

Keterangan:

x1 = *latitude* lokasi pencari kerja
 y1 = *longitude* lokasi pencari kerja
 x2 = *latitude* lokasi pekerjaan
 y2 = *longitude* lokasi pekerjaan

Dari penggunaan fungsi tersebut dapat dihitung berapa jarak dari titik koordinat pencari kerja ke lokasi pekerjaan yang ada di *database*. Jika jarak pekerjaan kurang dari radius yang telah ditentukan, maka pekerjaan akan dapat ditemukan pada aplikasi dengan ditandai dengan tanda berupa *marker* berwarna merah yang ada pada tampilan peta. Sedangkan untuk metode perhitungan manual dari mencari jarak berdasarkan radius dapat menggunakan formula Haversine, contoh perhitungannya sebagai berikut:

$$\begin{aligned} \Delta latitude &= latitude2 - latitude1 \\ \Delta longitude &= longitude2 - longitude1 \\ a &= \sin^2\left(\frac{\Delta latitude}{2}\right) + \cos(latitude1) * \cos(latitude2) * \sin^2\left(\frac{\Delta longitude}{2}\right) \\ c &= 2 * \text{atan}^2(\sqrt{a}, \sqrt{1-a}) \\ d &= R * c \end{aligned}$$

Gambar. 20 Formula Haversine[23]

Keterangan:

R = jari-jari bumi sebesar 6371(km)
 $\Delta latitude$ = besaran perubahan *latitude*
 $\Delta longitude$ = besaran perubahan *longitude*
 c = kalkulasi perpotongan sumbu
 d = jarak (km)

Contoh perhitungan formula Haversine dalam menghitung jarak antara 2 titik koordinat:

1) Tentukan titik koordinat pencari kerja dan lokasi pekerjaan. Titik Koordinat pertama adalah lokasi pencari kerja dengan *latitude*: -6.227415117322623 dan *longitude* 106.91186929914713. Titik Koordinat kedua adalah lokasi pekerjaan Cuci dan Setrika Pakaian 5Kg - 3Hari dengan *latitude*: -6.227621297222354 dan *longitude* 106.91200375556947 .

2) Mengubah nilai *latitude* dan *longitude* kedua titik menjadi radian.

Radian Latitude 1 = -6.227415117322623 * $\pi/180$ = -0.108688898 radian
 Radian Latitude 2 = -6.227621297222354 * $\pi/180$ = -0.108692496 radian
 Radian Longitude 1 = 106.91186929914713 * $\pi/180$ = 1.865964128 radian
 Radian Longitude 2 = 106.91200375556947 * $\pi/180$ = 1.865966475 radian

3) Tentukan $\Delta latitude$ dan $\Delta longitude$

$\Delta latitude$ = Latitude 2 – Latitude 1
 = -0.108692498 – (-0.108688898)
 = 0.00000362
 $\Delta longitude$ = Longitude 2 – Longitude 1
 = 1.865966475 - 1.865964128
 = 0.000002347

4) Tentukan nilai a:

$a = \sin^2\left(\frac{\Delta latitude}{2}\right) + \cos(latitude1) * \cos(latitude2) * \sin^2\left(\frac{\Delta longitude}{2}\right)$
 $a = \sin^2(0.00000362/2) + \cos(-0.108688898) * \cos(-0.108692496) * \sin^2(0.000002347/2)$
 $a = 0.00000000004605967864922$

5) Tentukan nilai perpotongan sumbu (c)

$c = 2 * a \sin(\sqrt{a}) = 2 * a \sin(\sqrt{0.00000000004605967864922})$
 $c = 0.0000042923$

6) Tentukan nilai jarak (d)

$d = R * c = 6371(\text{km}) * 0.0000042923 = 0.027346267 \text{ km.}$

Hasil perhitungan didapatkan jarak sebesar 0.027346267 km yang dihitung dari koordinat pencari kerja ke koordinat pekerjaan Cuci dan Setrika Pakaian 5Kg - 3Hari. Hasil tersebut kemudian dibandingkan dengan radius maksimal pekerjaan yang dipilih pada aplikasi, jika jarak kurang dari radius maksimal maka pekerjaan akan muncul ke tampilan peta pada aplikasi.

Pada tahap pengujian ini, dilakukan pembuatan 10 permintaan pekerjaan kedalam aplikasi sesuai dengan data yang dapat dilihat pada Tabel 1.

TABEL I
DATA PERMINTAAN PEKERJAAN

No	Nama Pekerjaan	Koordinat Lokasi Pekerjaan	
		Latitude	Longitude
1	Cuci dan Setrika Pakaian 5Kg - 3Hari	-6.22762129722354	106.91200375556947

No	Nama Pekerjaan	Koordinat Lokasi Pekerjaan	
		Latitude	Longitude
2	Merawat Kebun	- 6.22633610014 2662	106.911958493 29235
3	Membersihkan Gudang	- 6.22872151141 4059	106.912769190 96708
4	Mengasuh Anak Bayi	- 6.23100692322 3796	106.909393295 6457
5	Mencuci Mobil	- 6.23028433821 1875	106.893173307 1804
6	Cuci Baju Keluarga	- 6.21485017825 864	106.913841739 2969
7	Membersihkan Dapur	- 6.22462494334 0327	106.912938505 41115
8	Merawat Kebun	- 6.21836154196 8327	106.906350664 79445
9	Merapikan Lemari Buku	-6.2176426	106.8915948
10	Setrika Baju	-6.1826294	106.947666

Dengan posisi pencari kerja berdasarkan koordinat dari *Google Maps* ada pada posisi -6.227415117322623 *Latitude* dan posisi 106.91186929914713 *Longitude*. Berdasarkan koordinat tersebut, maka pekerjaan yang dapat ditemukan di aplikasi berdasarkan rentang radius ditunjukkan pada Tabel 2.

TABEL III
PERMINTAAN PEKERJAAN BERDASARKAN RADIUS

No Pekerjaan	Radius				
	<= 1 Km	<= 2 Km	<= 5 Km	<= 10 Km	<= 15 Km
1	Y	Y	Y	Y	Y
2	Y	Y	Y	Y	Y
3		Y	Y	Y	Y
4				Y	Y
5					
6					Y
7			Y	Y	Y
8					Y
9					
10					

Keterangan: Y = pekerjaan yang tersedia

Berdasarkan data tersebut, penggunaan radius dapat dimanfaatkan dalam memilih pekerjaan yang paling terdekat dari pengguna yakni pekerjaan nomor 1 dan 2, sedangkan aplikasi tidak akan menunjukkan pekerjaan nomor 5, 9 dan 10 dikarenakan jaraknya melebihi dari 15 Km dari pengguna.

Metode pencarian pekerjaan dengan metode radius yang dilakukan oleh pencari kerja memiliki kekurangan dalam hal titik koordinat pencari kerja, koordinat tersebut hanya mengambil lokasi pengguna pada saat mengakses aplikasi,

sehingga jika pengguna sedang berada di suatu tempat yang jauh dari rumahnya tetapi dia ingin mengambil pekerjaan yang berada di sekitar rumahnya, maka pencari kerja tersebut harus kembali menuju rumahnya dan melakukan pencarian pekerjaan dari titik lokasi tersebut dikarenakan radius maksimal yang diatur aplikasi hanya mencapai paling jauh 15 kilometer dari titik koordinat pencari kerja.

Tahapan pencarian pekerjaan secara sistem dilakukan dengan mengambil data pekerjaan yang berada di *database* dengan parameter *query* pekerjaan yang berstatus *Open*. Data pekerjaan tersebut kemudian dikirimkan melalui *backend* ke aplikasi dengan perantara *web service*. Setelah data sampai di aplikasi, data lokasi pekerja yang berupa *latitude* dan *longitude* akan dihitung jaraknya ke titik koordinat lokasi pencari kerja berada, kemudian pekerjaan yang kurang dari atau sama dengan nilai radius yang sudah ditetapkan pencari kerja, akan tampil pada peta dengan tanda *Marker* berwarna merah. Pekerjaan tersebut dapat diambil oleh pencari kerja dengan menekan marker lalu menekan tombol “Ambil Pekerjaan” untuk menyimpan pekerjaan tersebut sebagai pekerjaan yang akan dikerjakan oleh pencari kerja. *UI* hasil dari pencarian pekerjaan disekitar yang dilakukan oleh pencari kerja dapat terlihat pada Gambar 21.

Gambar. 21 Pekerjaan disekitar mitra

Pada aplikasi juga disediakan bantuan berupa nomor telepon pengguna yang membuat pekerjaan serta dapat diarahkan menggunakan layanan *Google Maps* ke tempat tujuan dari pekerjaan tersebut. Ketika pekerjaan yang berstatus *Open* sudah diambil oleh pekerja atau mitra, maka status akan berubah menjadi *Process* yang artinya dalam hal ini pekerjaan tersebut sudah tidak dapat diambil oleh pencari kerja lain dan harus diselesaikan oleh pencari kerja yang telah mengambil pekerjaan tersebut. Pemberi kerja yang permintaan pekerjaannya telah diambil oleh pencari kerja, akan mendapatkan notifikasi dengan informasi bahwa pekerjaan yang ia buat telah mendapatkan pekerja.

Notifikasi yang diterima pengguna dapat terlihat seperti pada Gambar 22.

Gambar. 22 Notifikasi aplikasi

Pada tahap selanjutnya merupakan tahapan pemrosesan pekerjaan yang telah diambil oleh pencari kerja yang dapat dilihat pada Gambar 23.

Gambar. 23 Activity Diagram pemrosesan pekerjaan

Pencari kerja disarankan untuk menghubungi pengguna untuk memastikan pekerjaan tersebut menggunakan nomor telepon yang telah disediakan dengan menekan tombol “Nomor Telepon”. Sesudah menghubungi pihak pemberi kerja, kemudian pencari kerja datang ke lokasi pengguna dengan menekan tombol “Dapatkan Arah” sehingga mitra diarahkan ke lokasi pengguna menggunakan bantuan *Google Maps* sesuai yang terlihat pada Gambar 21. Setelah

pencari kerja sampai di lokasi pemberi kerja dan menyelesaikan pekerjaan yang diminta, pencari kerja dapat menekan tombol “Pekerjaan Saya” kemudian pilih pekerjaan yang telah selesai, lalu tekan tombol “Selesai” seperti yang terlihat pada Gambar 24.

Gambar. 24 Daftar pekerjaan pencari kerja

Hal tersebut akan mengubah status pekerjaan menjadi *Finish* yang berarti bahwa pekerjaan telah berhasil diselesaikan oleh pencari kerja. Pekerjaan yang telah diselesaikan pencari kerja dapat ditutup oleh pemberi kerja yang akan mengubah kembali statusnya menjadi *Closed*.

Pada tahap akhir ketika pekerjaan sudah ditutup, pemberi kerja akan diminta untuk memberikan ulasan terhadap kinerja dari pencari kerja dalam bentuk *rating* berupa bintang dari rentang 1 sampai dengan 5. Metode *Rating Scale* adalah metode penilaian yang paling umum digunakan. Metode *Rating Scale* ini menggunakan skala untuk menilai kinerja dari pencari kerja yang telah menyelesaikan pekerjaannya. Pada skala *rating* ini bintang 1 menandakan bahwa pencari kerja bekerja dengan sangat buruk, bintang 2 berarti buruk, bintang 3 sedang, bintang 4 baik dan bintang 5 sangat baik. *Rating* yang diterima dari setiap pekerjaan yang telah diselesaikan akan dijumlahkan dengan semua *rating* dari seluruh pekerjaan yang telah didapatkan oleh pencari kerja, kemudian dihitung rata-ratanya dengan dibagi berdasarkan jumlah pekerjaan yang telah diselesaikan. Halaman pemberian ulasan dapat dilihat pada Gambar 25.

Pencari kerja yang mendapatkan nilai *rating* yang rendah secara berulang akan menjadi catatan dan pertimbangan oleh admin aplikasi dalam tetap mempertahankan pencari kerja dalam aplikasi atau pun menonaktifkan statusnya sehingga pencari kerja tersebut tidak dapat mengakses aplikasi.

Pada tahapan pengujian aplikasi, telah dilakukan pengujian menggunakan metode *Black Box* pada berbagai perangkat Android dengan hasil seperti pada Tabel 3.

Gambar. 25 Pemberian ulasan dan rating

TABEL IIIII
HASIL PENGUJIAN APLIKASI DENGAN BLACK BOX TESTING

Versi An-droid	Dapat Terpa-sang	Masuk dan Mendaftar	Membuat Pekerjaan	Mencari Pekerja-an	Mem-berikan Rating
11	√	√	√	√	√
10	√	√	√	√	√
9	√	√	√	√	√
8	√	√	√	√	√
7	√	√	√	√	√
6	×	×	×	×	×
5	×	×	×	×	×
4	×	×	×	×	×

Keterangan: √ = berhasil, × = gagal.

Pengujian dilakukan oleh 20 pengguna dengan berbagai macam versi Android, di mana aplikasi ini dapat berjalan di 19 perangkat pengguna tersebut sedangkan 1 pengguna tidak dapat menginstall aplikasi karena syarat minimum perangkat adalah Android versi 7. Hasil pengujian tersebut menunjukkan bahwa aplikasi pencarian pekerjaan jasa rumah tangga dengan metode radius dan *rating* berbasis Android dapat berjalan pada perangkat Android versi 7 sampai 11.

IV. KESIMPULAN

Penggunaan *Google Maps API* dapat diterapkan dalam pengembangan aplikasi yang membutuhkan pengambilan titik lokasi dengan memanfaatkan *Global Positioning System (GPS)*. Pada penelitian ini *Google Maps API* digunakan untuk mengambil titik lokasi pengguna yang kemudian akan dijadikan acuan dalam menghitung jarak terdekat dengan titik koordinat dari pencari kerja sehingga semua pekerjaan yang berada dalam jangkauan atau radius pencari kerja akan tampil pada peta di aplikasi. Pada penelitian ini aplikasi juga mengatur transaksi dari setiap pekerjaan yang dibuat hingga proses pekerjaan tersebut ditutup. Selain itu *User Interface (UI)* yang dinamis, keamanan pengguna dengan verifikasi kartu identitas, serta adanya penilaian terhadap kinerja pekerja diharapkan dapat mendorong minat masyarakat dalam penggunaan aplikasi ini agar membantu memperbanyak lapangan pekerjaan

rumah tangga sehingga membantu pekerja yang sedang membutuhkan sumber penghasilan tambahan. Kekurangan pada aplikasi ini dalam hal titik koordinat pencari kerja, di mana koordinat tersebut hanya mengambil lokasi pengguna pada saat mengakses aplikasi, sehingga pencari kerja tidak dapat mengambil pekerjaan jika posisinya berada jauh dari lokasi yang diinginkannya. Aplikasi ini dapat berjalan optimal pada perangkat berbasis Android dengan minimal versi 7 hingga versi 11. Hasil penelitian ini yaitu terciptanya aplikasi yang dapat membantu dalam memberikan lapangan pekerjaan jasa rumah tangga bagi masyarakat yang membutuhkan dengan melakukan pencarian pekerjaan terdekat dari lokasi pencari kerja ke lokasi pekerjaan. Aplikasi ini dapat berjalan dengan baik dengan akurasi 95% dari hasil uji coba terhadap 20 pengguna dimana 19 pengguna dapat memasang dan menjalankan aplikasi dengan baik sedangkan 1 pengguna tidak berhasil melakukan pemasangan dikarenakan versi Android dibawah versi 7.

REFERENSI

- [1] ILO, "COVID-19 and the world of work: Impact and policy responses," no. March, pp. 1–15, 2020.
- [2] International Labour Organization (ILO), "COVID-19 and the world of work," *Int. Labour Organ.*, no. April, pp. 1–22, 2020, [Online]. Available: https://www.ilo.org/wcmsp5/groups/public/@dgreports/@dcomm/documents/briefingnote/wcms_749399.pdf.
- [3] M. . Ir. Ismail Pakaya, "Sokong 9 Lompatan Kemnaker," *Bul. Bicara Data Ed. 1 Tahun 2021*, vol. 1, 2021, [Online]. Available: https://satudata.kemnaker.go.id/files/Draft_Buletin_2021_Edit_IS_SN_1620022659.pdf.
- [4] W. Marison, "Ketika Korban PHK akibat Corona Mendaftar Jadi ART untuk Bertahan Hidup," *Kompas.com*, 2020. <https://megapolitan.kompas.com/read/2020/05/05/14580741/ketik-a-korban-phk-akibat-corona-mendaftar-jadi-art-untuk-bertahan-hidup> (accessed Jun. 12, 2021).
- [5] A. Ramadhan, "Aplikasi Pencarian SMA/SMK DI Wilayah Jakarta Barat Berbasis Android," *J. SAINTEKOM*, vol. 9, no. 2, p. 119, 2019, doi: 10.33020/saintekom.v9i2.86.
- [6] APJII, "Buletin APJII," *Asosiasi Penyelenggara Jasa Internet Indonesia*, p. 1, 2020, [Online]. Available: <https://apjii.or.id/content/read/104/503/BULETIN-APJII-EDISI-74---November-2020>.
- [7] A. Eko, S. Putro, H. Tolle, and A. P. Kharisma, "Rancang Bangun Aplikasi Penawaran dan Pencarian Kerja Paruh Waktu (Part Time) Berbasis Lokasi," *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 8, pp. 2548–964, 2018, [Online]. Available: <http://j-ptiik.ub.ac.id>.
- [8] B. Yulianto and R. Layona, "Aplikasi Pencarian Tempat Wisata Berbasis GPS dengan Metode Radius dan Rating," *ComTech Comput. Math. Eng. Appl.*, vol. 6, no. 1, p. 109, 2015, doi: 10.21512/comtech.v6i1.2296.
- [9] R. Fauzi, S. Wibowo, and D. Y. Putri, "Perancangan Aplikasi Marketplace Jasa Percetakan Berbasis Website," *Fountain Informatics J.*, vol. 3, no. 1, p. 5, 2018, doi: 10.21111/fij.v3i1.1824.
- [10] N. Nurfadillah and N. R. Kurnianda, "Analisa dan Perancangan Aplikasi Pengolahan Sampah Rumah Tangga Berbasis Android," *J. Ilm. FIFO*, vol. 11, no. 1, p. 41, 2019, doi: 10.22441/fifo.2019.v10i1.005.
- [11] F.- Sonata, "Pemanfaatan UML (Unified Modeling Language) Dalam Perancangan Sistem Informasi E-Commerce Jenis Customer-To-Customer," *J. Komunika J. Komunikasi, Media dan Inform.*, vol. 8, no. 1, p. 22, 2019, doi: 10.31504/komunika.v8i1.1832.
- [12] M. Abror, A. Martono Dwi, and Aisah, "Rancang Bangun Aplikasi Sistem Informasi Antrean Bengkel Kendaraan Bermotor Berbasis

- Location Based Service Di Kota Malang,” *J. JARTEL*, vol. 9, no. 2, pp. 132–137, 2019.
- [13] F. Sandy and I. Chaidir, “Pemanfaatan GPS (Global Positioning System) Menentukan Posisi Autocare Terdekat Menggunakan Metode Sequential,” *J. Teknol. Elektro, Univ. Mercu Buana*, vol. 8, no. 2, pp. 131–137, 2017.
- [14] D. A. H. Capah and T. Herdi, “Penerapan Aplikasi Location Based Service Dalam Penanganan Gangguan Jaringan Berbasis Mobile,” *J. Inf. Syst. Informatics*, vol. 3, no. 1, pp. 135–143, 2021, doi: 10.33557/journalisi.v3i1.103.
- [15] R. Rizal and A. Rahmatulloh, “Restful Web Service Untuk Integrasi Sistem Akademik Dan Perpustakaan Universitas Perjuangan,” *J. Ilm. Inform.*, vol. 7, no. 01, p. 54, 2019, doi: 10.33884/jif.v7i01.1004.
- [16] T. Hidayat and M. Muttaqin, “Pengujian Sistem Informasi Pendaftaran dan Pembayaran Wisuda Online menggunakan Black Box Testing dengan Metode Equivalence Partitioning dan Boundary Value Analysis,” *J. Tek. Inform. UNIS JUTIS*, vol. 6, no. 1, pp. 2252–5351, 2018, [Online]. Available: www.ccssenet.org/cis.
- [17] D. Jagodić, Ž. Jovanović, S. Randić, and D. Vujičić, “Java Spring Boot Rest Web Service Integration With Java,” no. November, pp. 323–327, 2017.
- [18] A. Sonita and R. F. Fardianitama, “Aplikasi E-Order Menggunakan Firebase Dan Algoritme Knuth,” *J. Pseudocode*, vol. 5, no. 2, pp. 38–45, 2018.
- [19] E. A. W. Sanad, “Pemanfaatan Realtime Database di Platform Firebase Pada Aplikasi E-Tourism Kabupaten Nabire,” *J. Penelit. Enj.*, vol. 22, no. 1, pp. 20–26, 2019, doi: 10.25042/jpe.052018.04.
- [20] A. Febi, “Jurnal MNEMONIC REALTIME NOTIFICATION PADA APLIKASI BERBASIS WEB Realtime Notification pada Aplikasi Berbasis Web,” *Mnemonic*, vol. 1, no. 2, pp. 14–17, 2018.
- [21] M. S. Ardian Dwi Praba, “Studi Perbandingan Performansi Antara Mysql Dan Postgresql,” *J. Khatulistiwa Inform.*, vol. VIII, no. 2, pp. 1–6, 2020.
- [22] R. Munadi, Z. Musliyana, T. Y. Arif, A. Afdhal, and S. Syahrial, “Kombinasi Waktu Sinkronisasi dan Nilai Salt untuk Peningkatan Keamanan pada Single Sign-On,” *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 5, no. 3, 2016, doi: 10.22146/jnteti.v5i3.257.
- [23] A. M. Abdillah, R. Rianto, and N. I. Kurniati, “Penerapan Metode Haversine Pada Aplikasi Layanan Perbaikan Kendaraan Berbasis Location Based Service,” *JUITA J. Inform.*, vol. 7, no. 2, p. 81, 2019, doi: 10.30595/juita.v7i2.4141.